Amendments to the Specification:

Please amend the first paragraph of the detailed description, located on page 5, as follows:

An embodiment of the present invention provides machine automation objects implemented in a client-server model to control operation of one or more client machines from a single machine automation control module. A machine automation configuration, such as a test automation configuration, includes a server process, running in either a server or a client machine, and a client process, running in a client machine. The machine automation control module executes in the server process and initiates a machine automation server object. The machine automation control ~~modules~~ module specifies a given client machine and instructs the machine automation server object to create a machine automation client object in the client process on the specified client machine.

Please amend the first and second full paragraphs of page 35 as follows:

Under command of the machine automation control module, creating operation 302 instantiates the machine automation server object in the server process. Under command of the machine automation control module, identifying operation 304 provides a client machine identifier to the machine automation server object. Coupling operation 306 couples the server process to the client machine via a communication mechanism, such as DCOM (if the server process is running on a machine different than the client machine) or COM (if the server process is running in the same machine as the client machine). Instructing operation 308 instructs the machine automation server object to instantiate the machine automation client object on the identified client machine. In an embodiment of the present invention, ~~a~~ an instruction provided by instructing operation 308 is received in a command from the machine automation control module.

Creating operation 310 instantiates a machine automation client object on the ~~identifies~~ identified client machine via the communication mechanism (e.g., DCOM or COM). Under command of the machine automation control module, instructing operation 312 instructs the machine automation server object to call a method in the machine automation client object to control operation of the client machine.

Please amend table 1 of the specification, located on page 14, as follows:

Object, Function, or Property	Description
BTLog	The BTLog object includes methods for logging events, test conditions, test results (e.g., pass/fails), and other information generated during the machine automation process. This information, for example, is useful for reporting and interpreting test results in an automated testing system. In one embodiment, the BTLog ~~objects~~ object supports logging methods similar to those supported in an office automation system.
Init	The Init method accepts a machine identifier and causes a BTLog client object to be instantiated on the specified machine automation client.
LogClose	The LogClose method accepts a string comment parameter, which it appends to the end of the log. The method then finishes the logging process by closing the open log. The method returns a Boolean value indicating whether the method was successful.
LogGetLoggingFolder	The LogGetLoggingFolder method retrieves a pathname to the folder that is open and receiving the log information. The method returns a string identifying the current logging folder, or an empty string if the method fails.
LogOpen	The LogOpen method opens a log for the given object. The method is passed the name of the log, the log file name, a description of the log, and a comment for the beginning of the log file. The method returns a Boolean value indicating whether the method was successful.
LogSetOptions	The LogSetOptions method sets options for a new log. For example, an APPEND option may be set to indicate that a new log is to be appended to the end of the old log file by the same name (the default is that the old log is overwritten). The method returns a Boolean value indicating whether the method was successful.
LogStatus	The LogStatus method logs an entry into the open log file for a given object. The method is passed a string indicating the logging user; an enumerated value indicating the type of log event (e.g., Comment, Error, AppIdle, ASSERT, etc.); another enumerated value indicating success, failure, warning, or error; and a comment passed as a string. The method returns a Boolean value indicating whether the method was successful.

Table 1 – BTLog Object

Please amend table 4 of the specification, located on page 19, as follows:

Object, Function, or Property	Description
Images	The Images object is used to create and restore an image of a given disk of the machine automation client.
Init	The Init method accepts a machine identifier and causes an Images client object to be instantiated on the specified machine automation client.
PQDIArguments	The PQDIArguments property allows a user to specify arguments for an application (e.g., PDQI.exe) that creates or restores an image. ~~PDQI~~ PQDI refers to "Power Quest Drive Image", used in an embodiment of the present invention. Other disk image techniques are also contemplated within the scope of the present invention.
TimeOut	The TimeOut property specifies the maximum time to wait for the client machine to create or restore an image. By default, the property is set to 1200 seconds. If the create or restore image operation (e.g., PQDICreateImage or PQDIRestoreImage) has not completed within the specified time, the method returns FALSE to indicate that the method failed.
PQDICreateImage	The PQDICreateImage method allows a user to create a disk image of a client hard drive. The method accepts an image name string as an input parameter and creates a named image of the entire hard drive, including all partitions. The method automatically reboots to DOS (Disk Operating System), creates the named image on a network location, and reboots to the native OS (Operating System) of the client. The method returns a Boolean value indicating whether the method was successful.
PQDIRestoreImage	The PQDIRestoreImage method allows a user to restore a disk image of a client hard drive. The method accepts an image name string as an input parameter and restores the named image of the entire hard drive, including all partitions, thereby overriding the previous contents of the hard drive. The method automatically reboots to DOS, restores the named image from a network location to the client's hard drive, and reboots to the native OS of the client. The method returns a Boolean value indicating whether the method was successful.

Table 4 – Images Object

Please amend table 9 of the specification, located on pages 24-25, as follows:

Object, Function, or Property	Description
Snapshot	The Snapshot object provides functions for taking snapshots of a machine automation client's registry or associated files. The object can also generate differences between two snapshots. In one embodiment, a snapshot can include a textual file listing of all of or a subset of files located in any specified drive on a target computer. In an alternative embodiment, a ~~snap-shot~~ snapshot can include a list of registry keys and values found on a target computer.
Init	The Init method accepts a machine identifier and causes a Snapshot client object to be instantiated on the specified machine automation client.
DoDiff	The DoDiff method accepts two input file strings and an output file string as input parameters. The method outputs the differences between the two specified input snapshot files into the output file. The method returns a Boolean flag indicating whether the method was successful. Calculating a difference between two snapshots can produce a list of files or registry keys that were modified, added, or deleted between the times the snapshots were recorded.
DoDiffSubset	The DoDiffSubset takes a snapshot of a specified portion of a specified registry or associated files. The method returns a Boolean flag indicating whether the method was successful.
DoSnapshot	The DoSnapshot method accepts a file pathname string, a registry pathname string, an output file string, a file CRC Boolean value, and a "tokenize paths" Boolean flag as input parameters. The method takes a snapshot of a specified registry or file. The third parameter specifies the output file name. The fourth parameter specifies whether the user wants a file CRC generated. The file CRC can be used to identify changes in the snapshot file. The fifth parameter allows for the output file to be tokenized (e.g., "<<SYSDIR>>\oleaut32.dll". The method returns a Boolean flag indicating whether the method was successful.

Table 9 – Snapshot Object